Online Adaptation of Path Formation in UAV Search-and-Identify Missions

van Willigen, W.H.; Schut, M.C.; Eiben, A.E.; Kester, L.J.H.M.

published in
10th International Conference on Adaptive and Natural Computing Algorithms (ICANNGA)
2011

document version
Publisher's PDF, also known as Version of record

Online Adaptation of Path Formation in UAV Search-and-Identify Missions

W.H. van Willigen[1,2], M.C. Schut[1], A.E. Eiben[1], and L.J.H.M. Kester[2]

[1]VU University Amsterdam (NL)
[2]TNO Defence, Security and Safety, The Hague (NL)
{willem, mc.schut, ae.eiben}@few.vu.nl, leon.kester@tno.nl

Abstract. In this paper, we propose a technique for optimisation and online adaptation of search paths of unmanned aerial vehicles (UAVs) in search-and-identify missions. In these missions, a UAV has the objective to search for targets and to identify those. We extend earlier work that was restricted to offline generation of search paths by enabling the UAVs to adapt the search path online (i.e., at runtime). We let the UAV start with a pre-planned search path, generated by a Particle Swarm Optimiser, and adapt it at runtime based on expected value of information that can be acquired in the remainder of the mission. We show experimental results from 3 different types of UAV agents: two benchmark agents (one without any online adaptation that we call 'naive' and one with predefined online behaviour that we call 'exhaustive') and one with adaptive online behaviour, that we call 'adaptive'. Our results show that the adaptive UAV agent outperforms both the benchmarks, in terms of jointly optimising the search and identify objectives.

Keywords: adaptive algorithm; design and engineering for self-adaptive systems; unmanned aerial vehicles; search and identify.

1 Introduction

One of the most prevalent and important issues in reconnaissance, surveillance, and target acquisition (RSTA) flight missions is the ability to adapt one's flight path based on acquired information. In such (often military) missions, planes acquire information about a specific territory by first exploring it, followed by surveilling and finally obtaining information about possible targets in the area. While some information about the territory may be available beforehand (making a priori planning possible), it is increasingly important to do the planning during the mission itself because of the very dynamic nature of RSTA missions at present day (e.g., unknown territory, rapidly moving targets).

The possibility of such automated adaptability during the mission becomes very important when we take the human out of the loop, as we employ unmanned aerial vehicles (UAVs) in RSTA missions. The problem that we address in this paper concerns the programming of such UAVs in situations where some information is available beforehand (for example, some knowledge about possible target locations throughout the territory), but where substantial performance may be gained by equipping the UAVs with online (in-flight) adaptation of the flight path based on collected real-time information. We employ a machine-learning approach to accomplish this. Machine learning has been used to deal with different issues in UAV research and development. For example, Berger *et al.* [2] use a co-evolutionary algorithm for information gathering in UAV teams; Allaire *et al.* [1] have used genetic algorithms for UAV real-time path planning; and Sauter *et al.* [7, 5] have used a swarming approach (for which a ground sensor network for coordination purposes is needed).

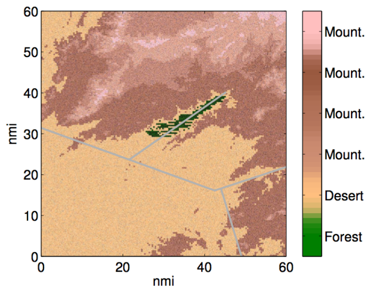
Recently, Pitre *et al.* [6] introduced a new measurement for (UAV) search and track missions. The introduced metric jointly optimises the objectives to 1) detect new targets, and 2) track previously detected targets. This particular metric has some desirable properties with respect to search-and-tracking: jointly optimises detection and tracking; easily compares different solutions; promotes early detection; encourages repeated observations of the same targets; and it is useful for resource management. However, this approach does not yet allow for online adaptation of the search path during the flight. In this paper, we provide a method for doing this. We build further on the work of Pitre *et al.* with two important differences: 1) we use the metric and calculations also for in-flight coordination and adaptation (whereas the original metric has reportedly only been used for off-line generation of paths), and 2) in our case study, the second objective (besides search) is to identify targets rather than tracking these.

This paper is structured as follows. In Section 2, we present the details of our adaptive algorithm. We report on the conducted simulation study in Section 3. Finally, Section 4 concludes and provides some pointers for future work.

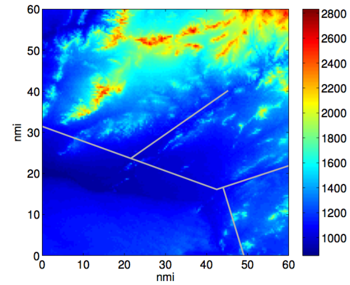
2 Model

In this section, we describe the model that we used in terms of (1) the problem setting (i.e., search-and-identification of targets in some terrain with UAVs), and (2) our solution approach (i.e., objective function and adaptive behaviour of the UAV). We describe both these aspects in detail below.

Our solution approach enables a UAV to jointly optimise the objectives of searching and identification by a UAV in a given terrain. Although we have no exact knowledge on where targets are in the terrain (because that would render the search-aspect of the mission pointless), we have some a priori knowledge in terms of probability distributions over the terrain cells on whether a target could be there. Before the mission, we compute an optimal flight path for the UAV. When the UAV is in-flight, it is possible to adapt this path. The before-mission calculation of the optimal search path as well as the in-flight decision to-adapt-or-not is based on a number of *value functions* that are described in detail below.



(a) Terrain types



(b) Altitude map

Fig. 1: Scenario Maps (Taken from [6]).

2.1 Problem setting

Terrain The terrain to-be-searched is 60 by 60 nautical miles (nmi). This consists of a mountainous area, a desert, a small forest and some roads. In Figures 1a and 1b, two maps of the terrain show the different types of terrain, and the different altitudes (that ranges from 856m to 2833m), respectively[1]. In both figures, the straight lines depict roads in the terrain.

A UAV that flies over the terrain cannot detect targets equally well in all types of terrain. We represent the ability-to-detect by means of a *detection probability*, denoted by p_{dot}, where *dot* means detection-on-terrain. In Table 1a, the detection probabilities for the different types of terrain are shown. The right column of this table shows that the detection probabilities increase when targets are on a road.

Table 1: Scenario Assumptions.

(a) Detection probabilities for different types of terrain.

	p_{dot}	p_{dot} on road
Desert	0.90	0.95
Mountain	0.5	0.75
Forest	0.10	0.50

(b) Percentage of targets per terrain type.

Terrain type	**% Targets**
Mountain	90
Desert	7
Road	2
Forest	1

Targets In this scenario, targets are stationary (i.e., non-mobile) objects located throughout the searched terrain. We consider all targets to be equally important

[1] These maps are the same that were used in [6].

(i.e., not prioritising with respect to a specific aim of a mission)[2]. Targets can be identified better when they are observed longer. We represent this gradually improving identification by means of a single scalar value, which increases as a UAV observes the object longer.

UAVs The UAVs in our model are planes that fly with a constant speed of 100 knots (kt) at a constant altitude of 3,000 meters above sea level. As previously mentioned, the UAV flies a particular search path that was determined beforehand. The adaptability of the UAV is that upon observation of a target, it *may* decide to fly a circle over the target enabling better identification. This decision depends on the objective function presented later in this section. After finishing the circle, it continues its original search path. A UAV has only limited resources (e.g., fuel), thus when it decides to fly a circle, this means that the path shortens in the tail (details follow below).

How much a UAV can see on the ground, depends on the altitude of the terrain. The detection range is defined as $range(alt) = -6.5 \cdot 10^{-4} \cdot alt + 1.96$, where alt is the altitude of the terrain. We assume a viewing angle of about 51 deg in every direction. In the lowest regions of the terrain, the detection range is 1.4 nmi, while in the higher regions, this number drops to just 0.1 nmi.

The probability that a UAV detects a target on the ground, denoted by $p_{det}()$, is determined by the detection range:

$$p_{det}(cell) = \begin{cases} p_{dot} & \text{if within } range(alt) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $cell$ is a single location in the terrain.

The UAV sensor automatically takes a picture every 30 seconds. In our scenario, a mission takes 2 hours, thus resulting in a total of 240 pictures taken and analysed. Finally, the maximum turning rate of the UAV is 2 degrees per second, which means that if the UAV wants to fly a circle above a certain object, this takes 3 minutes, or 6 pictures. Flying a circle above a target also means that the end of the search path is shortened by 3 minutes, or 6 pictures.

2.2 Solution Approach

We evaluate search paths by means of an objective function, based on (expected) *value functions*. This evaluation is needed for 1) the a priori calculations for determining optimal search paths, as well as for 2) in-flight adaptation of a search path. For the former (a priori search process), we provide more details in the following section. For the latter (in-flight adaptation), we provide details in this section after explaining the used value functions. We employ two different functions for evaluation: first, the value function, that computes the total value of a path *after* flying; and second, the expected value, that estimates the value

[2] In [6], extensions are introduced that allow for varying the target importance.

of a (partial) path *before* flying and, in case of the adaptive agent, *during* the flight.

The *value function* is defined as: $V = \sum_{t=1}^{T} \sum_{n=1}^{N} utilityGain(n,t)$, where T is the number of discrete time intervals during the mission, N is the number of detected targets at time t, and $utilityGain(n,t)$ is the gain in utility of information for target n at time t.

The utility gain function $utilityGain(n,t)$ can be interpreted as the number of points scored for observing a target. Upon first observation of a target, the utility gain is 1. This increases linearly with time for the duration of observation of this target with a maximum utility gain of 6 per target. The reason for this maximum is that identification cannot improve after 6 detections. However, after 6 consecutive non-detections (when a target seen before is now undetected), known information about that target is reset which means that when the UAV encounters that target after that time, new information can be gained yet again for that target.

We define the *expected value function* of a UAV search path as: $E(V) = \sum_{t=1}^{T} \sum_{c=1}^{C} p_{det}(c) p_{target}(c)$, where T is the number of discrete time intervals during the mission, C is the number of cells within the detection range of the UAV at time t, $p_{det}(c)$ is the probability of detection. This number depends on the type of terrain at cell c, and $p_{target}(c)$ is the probability of a target being present at cell c. We assume this information to be available and, because of the high resolution of the terrain, we also assume that no more than one target can be present at each cell.

This formula thus estimates the number of targets that will be detected during the length of the mission based on the probabilities of 1) the presence of a target and 2) detection by the UAV.

2.3 UAV adaptive agent

The UAV agent determines the behaviour of the UAV in terms of adapting the flight path or not. The *online adaptive agent* will decide on flying a circle above a detected target based on the *expected value of the remaining search path*. Pseudocode for this agent is depicted in Algorithm 1, that runs each timestep of the flight, when a picture has been taken.

When the UAV is currently not flying a circle (because otherwise the UAV could start flying circles within circles and this would increase the complexity of getting back on the original path significantly), and a new target has been observed, two values are computed: the expected value of the rest of the search path without flying a circle (expValueWithout), and the expected value of the rest of the search path with the certainty of observing a certain target during the circle (expValueWith).

3 Experiments

In this section, we describe the experimental design and setup, the results we obtained and an analysis of these results.

```
/* The UAV starts flying the predetermined search path. At each timestep t when a picture is
   taken and analysed, the following code is executed:                                    */
if the UAV detects a target that has not been seen before then
    /* Determine the expected value of the rest of the search path (from current timestep t
       until the final timestep T)                                                        */
    expValueWithout = E(V)_{t,T};
    /* Determine the expected value of the search path when a circle is made. To do this, the
       expected value gets 6 points for the circle (unless the expected value of the circle is
       greater than 6), and the rest of the path has been made 6 steps shorter.           */
    expValueWith = E(V)_{t+6,T} + max(6, E(V)_{t,t+5});

    if expValueWith > expValueWithout then
        flyCircle();
    else
        keepFollowingOriginalPath();
    end
end
```

Algorithm 1: The algorithm for the online adaptive UAV agent.

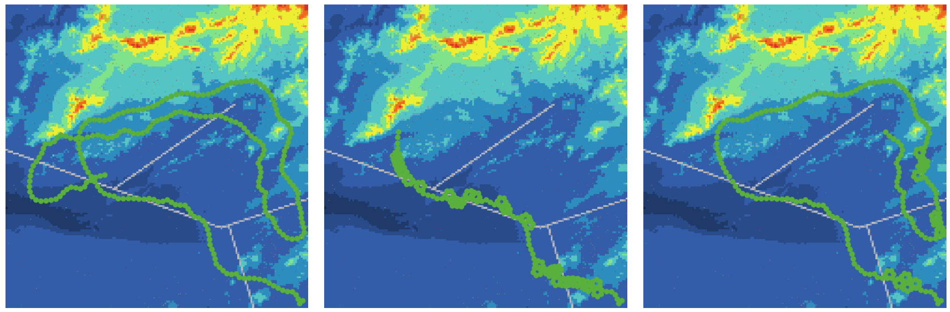
3.1 Design & Setup

The main objective of this research is to investigate if our online adaptive UAV agent improves the value of a predefined search path. To this end, we compare our agent, as described in section 2, to two benchmark agents: The *Naive Agent*, in which the UAV has a predefined search path and the UAV will just follow this path without doing anything differently. The *Exhaustive Agent* is the other benchmark and has predefined online behaviour: the UAV starts flying the predefined search path and each time the UAV detects a target, it always decides to fly a circle around that target before continuing its path. This agent is necessary in our experiments, because if we want to show that it is beneficial for the value to *sometimes* fly a circle, we also need to show that it is not a good idea to *always* fly a circle.

Our experimental design has 3 independent variables that we systematically vary to investigate the effects: 1) target distribution, 2) search path, and 3) agent type.

- *Target distributions:* We have generated 10 different target distributions, each consisting of 1,000 targets, placed in the terrain using the distribution as shown in table 1b. For each type of terrain, the targets are normally distributed.
- *Search paths:* We run the experiments on 10 different search paths. We generated search paths by hand and we ran a simple Particle Swarm Optimisation (PSO) technique [3] to optimise these search paths based on their expected value value. This work closely resembles the work described in [6]. After we ran the PSO algorithm for a fixed amount of time, we picked the 10 best paths for use in our experiments.
- *Type of agent:* As explained above, there are three types of agents: the naive agent (without any online adaptation), the exhaustive agent (that will always fly a circle upon detection of a new target) and the adaptive online agent (that will base its decision of flying a circle on expected value calculations).

The main measurable is the *obtained value* of a search path given a type of agent. The higher the value of a search path, the better. For each combination of



(a) (b) (c)

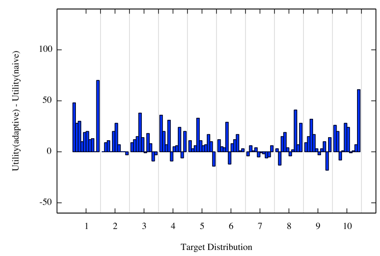
Fig. 2: (a) shows an example naive path, without online adaption; (b) shows an example exhaustive path, with many circles during the flight; and (c) shows an example adaptive path, with some circles here and there.

a search path and a target distribution, we measure the value of the paths that are generated by the three different agents. We hypothesise that the utilities of the paths generated by the adaptive agents are better than the utilities of the paths generated by the naive and the exhaustive agents. We also measure the number of *detected targets* and the total number of detections. Using these two metrics, we can see to what extent the different agents are better in searching, identification, or both.

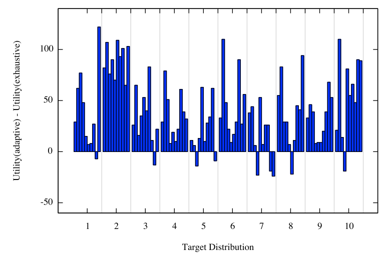
The different types of terrain and the detection probabilities of the different types of terrain were explained above in Section 2. The UAV starts flying in the bottom right corner of the world.

3.2 Results

Before we present the results of our simulations, we give some illustrative screenshots of the simulation, showing different kinds of search paths (albeit somewhat simplified for reasons of clarity). Here, the UAV starts in the bottom right corner of the terrain, and each green dot is a location at which the UAV takes a picture which is then analysed using one the three agents. An example flight is shown in Figure 2. In Figures 3a and 3b, the results for every run are shown in terms of value differences between the adaptive and the naive/exhaustive agents, respectively. On the x-axis of these charts are the 10 different target distributions. For all these 10 target distributes, the results for the 10 different search paths that we used are shown. On the y-axis, the difference in value is shown. Figures 4a and 4b are two histograms of the data from Figures 3a and 3b. From these histograms, it becomes clear that the data is not normally distributed, but slightly positively skewed. In the next section, we analyse this skewness. We also have included an example graph of this in Figure 5. The figure shows for each timestep that the value value of the search path up until that point. All lines are



(a) V(adaptive) - V(naive). Positive values mean that the adaptive agent has outperformed the naive agent.



(b) V(adaptive) - V(exhaustive). Positive values mean that the adaptive agent has outperformed the exhaustive agent.

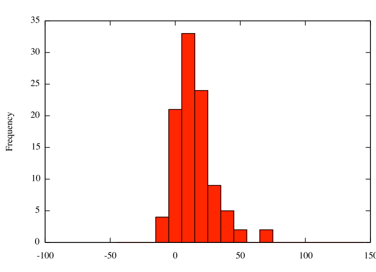
Fig. 3: Differences between the adaptive agent and the benchmark agents.

non-descending, since value will only increase over time. In table 2, the mean values for the total number of detections per run of the different agents is shown, as well as the mean number of uniquely detected targets per run. The ratio between these two values, which gives an indication on how well the identification objective is executed, is also included in this table.

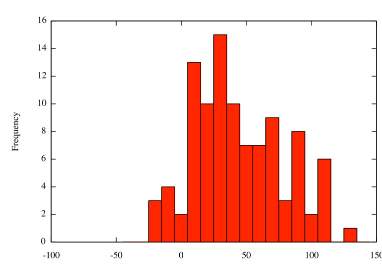
3.3 Analysis

From Figures 3a and 3b, we can see that the adaptive agent generally performs better than the naive method, and much better than the exhaustive method. Some exceptions occur, for instance distribution 7. We analysed these exceptions and these UAV paths do not encounter as many targets as expected.

The difference between the exhaustive and the adaptive agent are much larger. When many circles are flown in a short period of time, many targets



(a) Histogram of Value(adaptive) - Value(naive).



(b) Histogram of Value(adaptive) - Value(exhaustive).

Fig. 4: Histograms of the differences between the adaptive agent and the benchmark agents.

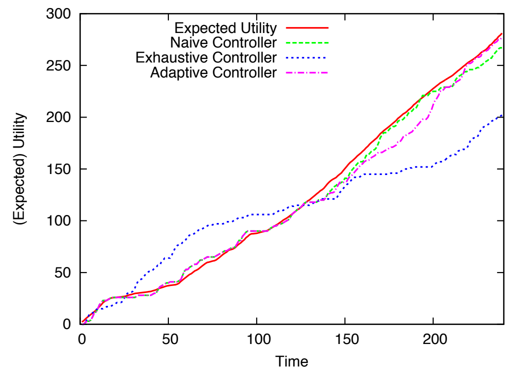


Fig. 5: Value increase over time (example).

Table 2: The mean values for the number of uniquely detected targets, the total number of detections and the ratio between these values.

	Naive	**Exhaustive**	**Adaptive**
# targets	171.04	68.44	148.99
# detections	315.04	362.22	347.09
detections / targets	1.84	5.29	2.33

will be detected for many more than 6 times, which yields no further utility gain. The histograms in Figure 4 are positively skewed. Using the Wilcoxon Signed-Rank test, we found that the adaptive agent is significantly better than the naive and exhaustive agents using a significance level of $p = 0.05$, which validates our hypothesis. Figure 5 depicts an example run. In this Figure, we observe that the naive agent does not significantly differ from the expected value. The exhaustive agent starts out well, but is outperformed by the other agents after some time. Note that Figure 5 is an example of one single run. Plots of other runs look differently. This can also be derived from the other plots; sometimes the naive or exhaustive agents are better. But generally, the plots follow this pattern.

Our second metric, i.e., the number of detections versus the number of uniquely detected targets, is depicted in Table 2. Using the numbers from this table, we can say something about strengths and weaknesses of each agent. We expected the naive agent to be the best in searching, the exhaustive agent to be the best in identification and the adaptive agent to be the best in jointly optimising these objectives. The naive agent has the highest mean number of uniquely detected targets, while the exhaustive agent has the highest ratio between the number of detections and the number of targets. The adaptive agent is best in jointly optimising these objectives.

4 Conclusions

In this paper, we propose a UAV agent that online adapts its predefined search path according to actual observations during the mission. The adaptive agent flies a circle above a detected target when it expects that this will improve the total value of the search path.

Our results show that our agent significantly outperforms both a naive and an exhaustive agent. However, not in every instance the adaptive agent outperforms the other two; in some cases one of the benchmarks is better. This result can be attributed to unexpected situations during the flight.

We also conclude that each agent has its own strength. It depends on the user's goal which agent is best. In our scenario, we want to jointly optimise search and identification objectives. Using these objectives jointly, our adaptive agent outperforms the benchmarks. But if searching was the only objective, the naive agent would be better; likewise, when identification was the only objective, the exhaustive agent would be the better one.

As a future research path, we will generalise the model further by introducing different kinds of vehicles with different kinds of capabilities (e.g., helicopters, ground vehicles, underwater vehicles). We will investigate how to model different capabilities and how the different vehicles in the field can make use of other vehicle's capabilities. Related work in this direction has been done by Kester *et al.* [4] to find a unifying way of designing *Networked Adaptive Interactive Hybrid Systems*.

References

1. Francois C. J. Allaire, Mohamed Tarbouchi, Gilles Labonte, and Giovanni Fusina. Fpga implementation of genetic algorithm for uav real-time path planning. *Intelligent Robot Systems*, 54:495–510, 2009.
2. Jean Berger, Jens Happe, Christian Gagne, and Martin Lau. Co-evolutionary information gathering for a cooperative unmanned aerial vehicle team. In *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pages 347–354, 2009.
3. R. Eberhart and J. Kennedy. Particle swarm optimization. In *Proceeding of IEEE International Conference on Neural Networks*, volume 4, 1995.
4. Leon J. M. H. Kester. Designing networked adaptive interactive hybrid systems. In *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 516 – 521, August 20-22, 2008.
5. François Legras, Arnaud Glad, Olivier Simonin, and François Charpillet. Authority sharing in a swarm of UAVs: Simulation and experiments with operators. In *SIMPAR '08: Proceedings of the 1st International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, pages 293–304. Springer-Verlag, 2008.
6. Ryan R. Pitre, X. Rong Li, and Donald DelBalzo. A new performance metric for search and track missions 2: Design and application to UAV search. In *Proceedings of the 12th International Conference on Information Fusion*, pages 1108–1114, 2009.
7. John A. Sauter, Robert Matthews, H. Van Dyke Parunak, and Sven A. Brueckner. Performance of digital pheromones for swarming vehicle control. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 903–910, New York, NY, USA, 2005. ACM.